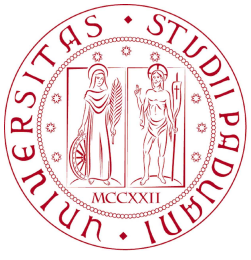
Università degli Studi di Padova
Facoltà di Ingegneria



UNIVERSITÀ DEGLI STUDI DI PADOVA
FACOLTÀ DI INGEGNERIA

—

DIPARTIMENTO DI INNOVAZIONE MECCANICA E GESTIONALE

—

TESI DI LAUREA TRIENNALE IN INGEGNERIA INFORMATICA

CELLA ROBOTIZZATA

Ottimizzazione di software di gestione

RELATORE: CH.MO PROF. GIULIO ROSATI

LAUREANDO: LAAMOURI HAMZA

ANNO ACCADEMICO 2010-2011

A mia moglie HIND

Finito di scrivere il giorno 22 marzzo 2011

"Il più GRANDE PIACERE nella VITA è fare quello che la gente dice che non sai fare."

Walter Bagehot

Indice

Sommario

Questa tesi si inserisce all'interno di un progetto finalizzato alla realizzazione di una cella robotizzata per l'assemblaggio di scambiatori di calore a pacco alettato in sviluppo presso il DIMEG (*Dipartimento di Innovazione Meccanica e Gestionale*) dell'Università di Padova.

Il progetto prevede lo studio di fattibilità e la prototipazione della cella automatizzata, con particolare riferimento al montaggio delle curve di chiusura del circuito idraulico sulla sommità dei tubi del pacco alettato seguendo percorsi ottimali che minimizzino il tempo di esecuzione del ciclo complessivo. La cella comprende un sistema di visione artificiale per l'individuazione dei centri-tubo e la rilevazione di eventuali difetti di mandrinatura, un robot SCARA preposto all'assemblaggio del pezzo e un software di gestione della cella.

L'attivita di studio svolta si sofferma sugli aspetti riguardanti la progettazione e l'implementazione del software di gestione della cella industriale, il quale si interfaccia con il sistema di visione, il controllore di un manipolatore industriale e il gestionale dell'azienda.

Introduzione

L'automazione, fin dalla sua introduzione nell'ambito industriale, si pone come obiettivo quello di ridurre al minimo l'intervento dell'uomo all'interno dei processi produttivi. I vantaggi legati alla sostituzione dell'operatore umano con un sistema di lavoro automatizzato si esprimono in termini di maggior velocità di lavorazione, maggior affidabilità e ripetibilità nei compiti assegnati. Tutto ciò si traduce in minori costi per l'impresa nel corso dell'esercizio produttivo ed un elevato e costante standard qualitativo dei prodotti finiti. Il ricorso all'automazione consente inoltre di portare a termine compiti altrimenti impossibili per gli esseri umani, come la gestione di carichi pesanti, l'interazione con sostanze pericolose o lavorazioni per le quali è richiesta un'elevata precisione.

Con l'avanzare dello sviluppo tecnologico, si allarga di conseguenza anche il campo dei processi industriali candidati alla conversione in senso automatizzato: avviene così che lavorazioni ritenute finora esclusivamente manuali divengano oggetto di studio, aprendo così nuove opportunità di ottimizzazione per le relative catene produttive. L'utilizzo dei sistemi di visione artificiale e di altri sistemi di percezione uniti a manipolatori industriali consentono di aumentare la versatilità e la sfera di applicazione dei sistemi automatizzati, permettendo a queste macchine di interagire con l'ambiente che le circonda in modo "intelligente" oltre che preciso.

Un processo produttivo di tale genere, non ancora completamente automatizzato nel settore industriale del condizionamento dell'aria, è quello relativo all'assemblaggio di scambiatori di calore a pacco alettato. Per scambiatore di calore intendiamo una batteria di circuiti tubolari che, inserita in un pacco di alette sovrapposte, consente di realizzare lo scambio termico tra un fluido gassoso esterno ed un fluido primario interno. Tale prodotto trova naturale appli-

cazione in campo civile e industriale all'interno di molti sistemi di raffreddamento o riscaldamento (condizionatori, chiller, ventilconvettori).

Attualmente il gruppo Mechatronics, del Dipartimento di Innovazione Meccanica e Gestionale (DIMEG) di Padova, sta finalizzando questo progetto di cella robotizzata che si inserirà in una particolare fase del processo di costruzione di scambiatori di calore a pacco alettato: l'inserimento delle curve di collegamento tra i tubi nella parte superiore dello scambiatore. L'interesse verso questo processo deriva dal fatto che attualmente, presso l'azienda committente così come in altre realtà aziendali, tale operazione viene ancora svolta manualmente da un operatore umano. Lo scopo è pertanto quello di predisporre un sistema robotizzato che una volta identificato correttamente la posizione dello scambiatore, provveda all'inserimento delle curve secondo uno schema prestabilito.

Questa tesi di laurea triennale in ingegneria informatica presenta e discute la parte del progetto riguardante la progettazione del sistema di gestione della cella automatizzata. Esso considera lo sviluppo del software in ambiente Matlab per l'interazione tra il sistema di visione artificiale deputato al riconoscimento e all'analisi dei singoli scambiatori posti sul piano di lavoro della cella, il codice relativo alla movimentazione del robot e l'intero corredo sensoristico ed elettrico di una tipica automazione.

Il **primo capitolo** é introduttivo al lavoro svolto e presenta le specifiche del progetto nella sua globalità. In particolare vengono analizzate le caratteristiche tipiche degli scambiatore di calore prodotti dall'azienda committente e descritta l'attuale linea di produzione. Viene poi presentato il progetto della cella per l'inserimento automatizzato delle curve, indicandone i requisiti e le specifiche di progetto che ne hanno determinato in linea di massima la struttura generale della cella.

Con il **secondo capitolo** si propone di descrivere il problema del percorso minimo in generale dando alcune soluzione ed algoritmi al problema specifico del comesso viaggaitore TSP , analisando il costo computazionale e la sua applicazione nel nostro caso si studio.

Il **terzo capitolo** descrive in dettaglio la parte software della gestione della cella , cita le strutture dei dati utilisate ed analisa in oltre gli algoritmi proposti per il calcolo del percorso del robot ottimezando il costo totale e elabora gli imagine dei scambiatori evedenziando il percorso seguito dal Robot .

Il **quarto capitolo** riporta tutti i risultati in tabelle e in imagine dei test fatti su scambiatori diversi, facendo il confronto tra i algoritmi proposti, che finsce con delle considarazione e conclusioni.

Ed ora, buona lettura!

Capitolo 1

Il progetto della cella

Il progetto di cella di lavoro robotizzata per l'assemblaggio di scambiatori di calore nasce dalla commissione di una ditta specializzata nella produzione di scambiatori di calore a pacco alettato per applicazioni sia civili che industriali.

1.1 Lo scambiatore di calore

Lo scambiatore di calore a pacco alettato (Figura 1.1) é una apparecchiatura che consente lo scambio termico tra aria e fluidi primari come l'acqua, gas frigorigeni od altro.

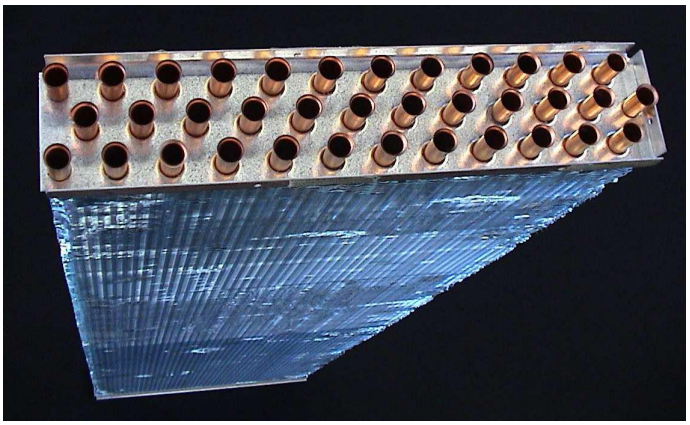


Figura 1.1: Scambiatore di calore.

Lo scambiatore é costituito principalmente da due elementi: un pacco alettato in alluminio o rame abbinato ad uno o più circuiti di tubi in rame montati nel pacco. Il fluido scorre all'interno dei tubi di rame collegati in modo tale da costituire un circuito chiuso. Questo circuito é prestabilito e studiato al fine di

avere il massimo scambio termico tra il fluido all'interno del percorso e il fluido all'esterno del percorso.

Il circuito di tubi di rame inserito nel pacco alettato viene realizzato utilizzando le cosiddette "forcine" (Figura 1.2), che non sono altro che tubi di rame piegati a "U" tramite un processo automatizzato. La forcina, lunga quanto lo scambiatore, viene inserita attraverso il pacco alettato e la piastra soprastante secondo schemi prestabiliti. Le forcine di un singolo circuito così inserite nel pacco vanno a costituire sulla superficie della piastra dello scambiatore un determinato **schema-tubi**, caratterizzato dal relativo numero, diametro dei tubi e da una ben precisa geometria dei fori di inserimento delle forcine nel pacco.

Figura 1.2: Forcine.

Gli schemi-tubi presi in considerazione in questo studio sono costruiti con tubi di rame aventi il diametro esterno di 9.52mm (3/8") oppure 7.94mm (5/16") e gli interassi tra i tubi sono descritti dalle seguenti due geometrie : 25mm × 21.65mm (geometria equilatera) e 25mm × 19mm (geometria non equilatera). Nello scambiatore visto dall'alto é possibile distinguire un ordinamento dei tubi per righe, i cosiddetti ranghi; ad esempio lo scambiatore visibile nella figura 1.1 presenta uno schema-tubi suddiviso in tre ranghi.

Una volta inserite le forcine nel pacco alettato, per chiudere il circuito vengono utilizzate le curve. La curva (Figura 1.3) é un tratto di tubo di rame piegato a "C" avente due tratti rettilinei alle estremità e di diametro esterno coincidente con quello delle forcine pre-mandrinate. Questo oggetto funge da collegamento tra una forcina e l'altra e rappresenta proprio la parte di tubo che chiude il circuito; posizionando infatti le curve secondo schemi prestabiliti e facendo in modo che

colleghino sempre forcine distinte si possono creare svariati circuiti chiusi che possono avere un ingresso e un'uscita oppure più ingressi e più uscite, a seconda dei tubi lasciati aperti.

Figura 1.3: Curve con e senza anello di saldatura.

Le curve possono essere provviste o meno di un anello di rame abbracciato attorno alle due estremità. L'anello, se presente, serve per gli scambiatori che vengono saldati in maniera automatizzata: l'anello infatti costituisce il materiale d'apporto per la saldo-brasatura. La saldobrasatura avviene manualmente se la curva é priva di anello, automatica se la curva ha l'anello. Se é prevista la saldatura automatizzata lo scambiatore viene costruito con la piastra dove

vengono alloggiati i bicchieri in lamiera zincata e non in lega di alluminio. Le curve presentano una superficie liscia e completamente a sezione circolare, quindi molto difficile da manipolare.

Le curve possono avere diversi diametri e diversi interassi. Quelle che vengono prese in considerazione in questo studio sono di soli due tipi:

- interasse $= 25mm$ (utilizzabile in tutte le geometrie) con o senza anello
- interasse $= 22.743mm$ (solo per geometria $25mm \times 19mm$) con o senza anello

Lo schema per mezzo del quale vengono posizionate le curve che collegano una forcina all'altra é presente nel disegno CAD (Figura 1.4) dello specifico scambiatore di calore. Ogni scambiatore di calore possiede i propri disegni costruttivi e i disegni contenenti gli schemi di montaggio delle forcine e delle curve (disegni CAD del tipo ST).

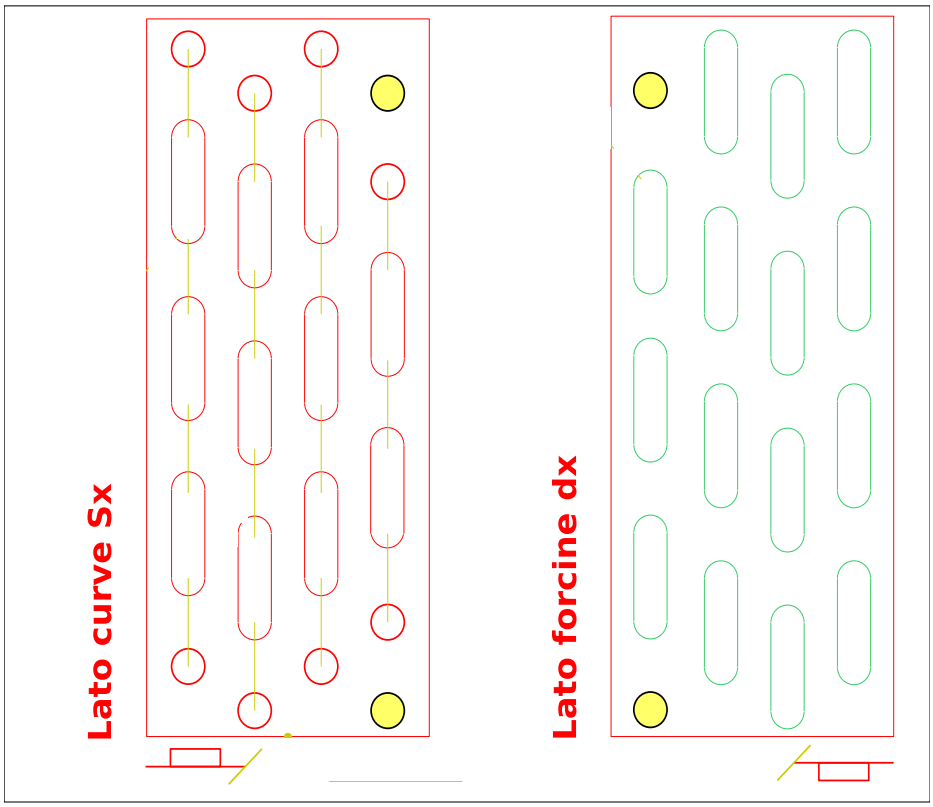


Figura 1.4: Disegno CAD di uno schema tubi a 4 ranghi.

Il bicchiere (Figura 1.5), rappresenta la sede di inserimento della curva nella forcina. Esso viene creato per mandrinatura. Sulla sommità del bicchiere viene realizzata una svasatura sul bordo con un duplice scopo: facilitare l'inserimento delle curve e soprattutto creare una zona di accumulo del materiale di apporto per la saldobrasatura delle curve sui bicchieri. La presenza della svasatura é molto utile in quanto costituisce un invito per il movimento di inserimento della curva.

Figura 1.5: Bicchieri.

L'operazione di mandrinatura é però responsabile di numerose tipologie di difetti accidentali. In primo luogo la flessione della barra, proporzionale alla sua lunghezza, durante l'inserimento all'interno del tubo può causare un allargamento del bicchiere del tubo non uniforme correlato ad una conseguente svasatura non uniforme. Dall'altro lato una variazione di altezza delle forcine, o della loro sporgenza dalla piastra, può signicare una svasatura più o meno accentuata e nel primo caso, stressando eccessivamente l'imboccatura in rame, può portarla ad aprirsi generando dei tagli sulla svasatura (Figura 1.6). L'entità delle variazioni di altezza dei bicchieri é nell'intorno dei $2.5mm \div 4mm$, a differenza dei disegni costruttivi indicanti tolleranze di $\pm 1mm$ su tutte le quote del bicchiere, compresa la sporgenza dalla piastra. Questa problema non influenza gli inserimenti, tanto quanto le altre tipologie di difetti.

Figura 1.6: Taglio sulla svasatura di un bicchiere.

Si rende inoltre necessario considerare la possibilità che la mandrinatura avvenga correttamente entro i valori previsti ma che successivamente le imboccature delle batterie, particolarmente delicate visto lo spessore del rame impiegato,

vengano danneggiate da colpi accidentali durante la manipolazione o il trasporto. Tutte le considerazioni avanzate sono realmente importanti, dal momento che dalla qualita dell'operazione di mandrinatura dipende il successo della seguente operazione di inserimento delle curve. Le diverse tipologie di difetti e le relative caratteristiche saranno discusse in modo più approfondito nei prossimi capitoli.

1.2 L'attuale linea di produzione

La catena di produzione di uno scambiatore di calore a pacco alettato può essere rappresentata dallo schema di figura 1.7, in cui sono distinte le varie lavorazioni in base all'ordine di esecuzione, ai loro input ed output. La fase di lavorazione oggetto di questo studio é stata evidenziata al centro del diagramma.

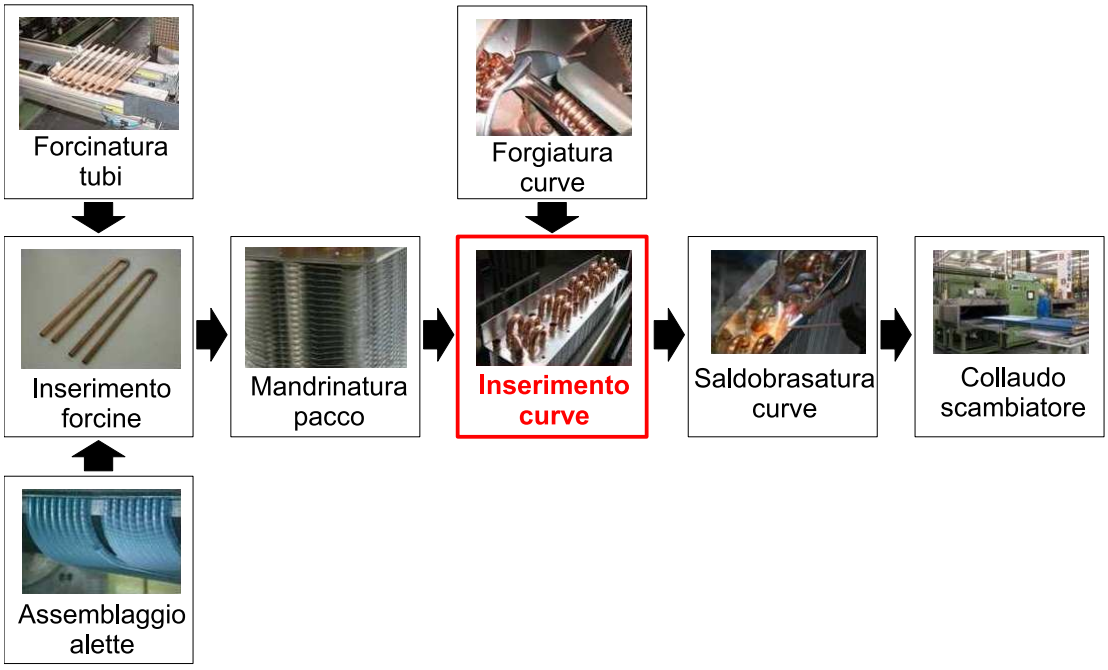


Figura 1.7: Ciclo di produzione di uno scambiatore di calore.

Il pacco alettato viene assemblato a partire da una pila di sottili lamine continue (alette), delimitata superiormente ed inferiormente da due piastre forate (in alluminio o acciaio zincato) eventualmente dotate di spalle laterali. Le alette sono anch'esse forate secondo la stessa geometria dei fori delle piastre delimitanti, ed in corrispondenza dei fori sono munite di un collare che permette di montarle

sovrapposte e ne definisce la distanza tra una e l'altra. I tubi che trovano alloggio all'interno del pacco alettato sono inseriti a coppie in forcina attraverso i fori sulla piastra inferiore del pacco. L'inserimento delle forcine avviene secondo lo schema-tubi descritto nel disegno CAD relativo al particolare scambiatore prodotto. L'accoppiamento tra le forcine e le alette nel pacco viene realizzato per interferenza tramite mandrinatura: un'espansione meccanica del tubo che si ottiene introducendovi un'ogiva. In questa fase di lavorazione rientra anche l'operazione di svasamento dell'imboccatura dei tubi nella parte superiore del pacco. L'operazione di mandrinatura ed i suoi effetti sullo scambiatore sono analizzati in dettaglio piu avanti.

La circuitazione nella parte superiore del pacco viene garantita dal collegamento di imboccature adiacenti con alcune curve in rame. Naturalmente la possibilita di inserimento delle curve dipende direttamente dal risultato della mandrinatura precedente. Le curve sono successivamente saldate ai tubi attraverso l'aggiunta di un quantitativo di rame a temperatura di fusione inferiore a quella dei tubi (saldobrasatura, Figura 1.8); tale materiale può essere prevelato da un'astina esterna o da uno specifico anello di brasatura gia montato sulle curve. Lo scambiatore viene quindi completato con l'inserimento dei collettori di alimentazione e scarico del fluido, aggiunte nella parte superiore della batteria sempre attraverso saldobrasatura. La tenuta della batteria cosi realizzata viene successivamente collaudata insufflandovi aria secca o elio e vericando la presenza di eventuali perdite con l'immersione in acqua.

Figura 1.8: Operazione di saldobrasatura delle curve.

Allo stato attuale la procedura di inserimento delle curve avviene in forma

manuale. Un operatore addetto alla catena di montaggio sistema lo scambiatore su di un nastro trasportatore fissandola alla base dal lato delle forcine. Tale nastro conduce molto lentamente i pezzi verso l'area dove avviene la successiva brasatura; in questo lasso di tempo l'operatore, curve alla mano, procede al loro inserimento nella parte superiore secondo lo schema di circuitazione prefissato dopo di che, concluso l'inserimento, ribatte ogni curva nella sua sede con un martello di gomma per migliorare l'accoppiamento con i tubi. A questo punto un'altro scambiatore può essere posizionata sul nastro e l'operazione si ripete. Una lavorazione di questo tipo ha ovviamente dei limiti e degli inconvenienti:

1. bassa velocità dell'operazione di montaggio;
2. necessaria separazione delle operazioni di inserimento e battitura;
3. possibilità di errori nell'inserimento;
4. rilevamento impreciso dei possibili difetti delle imboccature;
5. determina un carico di lavoro ripetitivo per l'operatore, dannoso sia per esso che per il processo (maggior probabilità d'errore);

Per quanto riguarda il primo punto è stato valutato sperimentalmente che in media il tempo impiegato da un'operatore che sappia esattamente dove posizionare la curva è di circa 2 secondi. Chiaramente per definire il tempo-curva, dato dal rapporto tra il tempo totale di lavorazione e il numero di curve da inserire, deve essere sommato al tempo di inserimento il tempo per la battitura. Pertanto la velocità complessiva dell'operazione risulta scarsa. Possiamo poi ipotizzare che un'operatore allenato ed esperto sappia sempre mettere le curve nelle sedi corrispondenti allo schema di circuitazione previsto, tuttavia la probabilità di un errore esiste ed è diversa da zero. Va invece a favore dell'operatore la possibilità di manipolare con facilità curve di dimensioni e diametro del tubo diverse. I possibili difetti alle imboccature dei tubi, che come si è detto rappresentano un frequente risultato dell'operazione di mandrinatura, sono sicuramente identificabili sommariamente dall'operatore (peraltro già impegnato a montare le curve) ma non qualitativamente. Un difetto grave può infatti rendere vana la brasatura

prefigurando una perdita nel circuito. Infine c'è da considerare l'importante fattore umano: una persona chiamata a svolgere un'operazione manuale ripetitiva, quindi priva di stimoli, è soggetta ad un aumento dello stress da lavoro. Tutte queste valutazioni vanno a favore della sostituzione dell'attuale modalità di lavorazione con una automatizzata, realizzata da una cella composta da un sistema di manipolazione delle curve (il braccio dell'operatore) e un sistema di visione artificiale che guidi il loro inserimento (l'occhio e il cervello).

1.3 La cella in progettazione: linee guida

Lo scopo dell'intero progetto, come già accennato, è la realizzazione di una completa cella di lavoro robotizzata in grado di inserire autonomamente e intelligentemente le curve nello scambiatore, finalizzandone la struttura e le funzionalità definitive. Questa tesi si è inserita nella fase di sviluppo intermedio dell'intero progetto, il quale si è articolato tra studio iniziale, prototipazione, sviluppo software e visione fino alle sperimentazioni e alla posa in opera.

Tale cella robotizzata é rappresentabile come un sistema che ha come input il prodotto da assemblare corredato dei relativi disegni CAD contenenti le specifiche di assemblaggio, e come output, il prodotto assemblato con una serie di dati relativi all'operazione di assemblaggio effettuato (Figura 1.9). Tra questi dati , ad esempio, vi sono i dati statistici, come il numero di pezzi assemblati e non assemblati a causa di difetti rilevati dal sistema di visione, quali e quanti di questi difetti sono stati rilevati, e la gestione degli ordini.

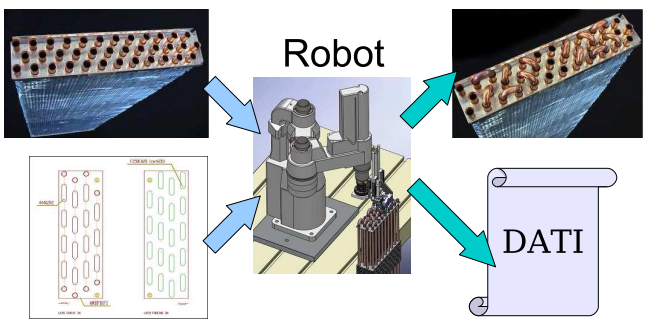


Figura 1.9: Schema di Input-Output della cella robotizzata.

1.3.1 Obiettivi del progetto

Il lavoro di tesi descritto nei successivi capitoli costituisce modifica ed ottimizzazione del software di gestione della cella, che si riassumono nella creazione del percorso del robot tramite le sue coordinate cartesiani e angolare basando su algoritmi di ottimizzazione del percorso e modificarli secondo l'esigenza del nostro problema.

Capitolo 2

I Problemi di riferimento

In questo capitolo andremo a analizzare la questione principale del percorso minimo, citando alcuni algoritmi che risolvono questo punto in generale. In particolare tratteremo sulle problematiche del commesso viaggiattore TSP(Travelling Salesman Problem) facendo poi riferimento al nostro problema primario. In ultima analisi, affroteremo il perchè, nel nostro caso e in relazione al problema centrale affrontato, queste soluzioni non possano essere adottate.

2.1 Cammino minimo (Shortest path)

Lo shortest path è, nella teoria dei grafi, il cammino minimo tra due vertici, ossia quel percorso che collega due vertici dati e che minimizza la somma dei costi associati all'attraversamento di ciascun lato.
Più formalmente il problema dello shortest path si può enunciare così: dato un grafo soppesato (cioè un insieme V di vertici, un insieme E di lati e una funzione che restituisca il costo sottoforma di numero reale: f: E → R) e dato inoltre un elemento (vertice) v di V, trovare un cammino P da v ad un altro distinto v' di V in modo che

$$\sum_{p \in P} f(p)$$

Sia la minima tra tutte quelle relative ai cammini che collegano v a v' . Il problema dello shortest path per tutte le coppie è simile. In tal caso bisogna trovare percorsi siffatti per ogni coppia di vertici v-v'.

Una possibile soluzione alla questione da noi presa in analisi dello shortest path è quella che viene chiamata un "algoritmo di tracciamento (di rotta)" (pathing algorithm). I più importanti algoritmi di questa categoria sono :

Algoritmo di Dijkstra -risolve problemi con una sola sorgente se tutti i pesi degli archi sono maggiori o uguali a zero. Senza richiedere un elevato tempo d'esecuzione, questo algoritmo può infatti calcolare la strada più breve da un determinato nodo di partenza "p" e tutti gli altri nodi del grafo.

Algoritmo di Bellman-Ford - risolve problemi con una sola sorgente, anche se i pesi degli archi sono negativi

Algoritmo di ricerca A* - risolve problemi con una sola sorgente usando l'euristica per tentare di velocizzare la ricerca.

Algoritmo di Floyd-Warshall - risolve tutte le possibili coppie.

Algoritmo di Johnso - risolve tutte le coppie, può essere più veloce dell'algoritmo di Floyd-Warshall su grafi sparsi.

Un problema simile a questo è quello del comesso viaggiatore in cui si cerca il percorso più breve che attraversi tutti i nodi del grafo una sola volta, per poi ritornare al punto di partenza. Questo problema è NP-Completo, per cui una soluzione efficiente potrebbe non esistere .

Il TSP o commesso viaggiatore é la soluzione megliore al nostro problema perchè in realità noi non cerchiamo il percorso minimo tra due nodi ma un percorso breve che attraversi tutti i nodi.

Ma, cosiderato l'elevato costo computazionale della soluzione sopra proposta, andremo a proporne una ulteriore. Per fare ciò ci baseremo sull'algoritmo sopra elencato, ovvero quello di Dijkstra, per poi svilupparne uno in relazione alle nostre esigenze da soddisfare.

2.2 Il problema del commesso viaggiatore (TSP)

Il problema del commesso viaggiatore o TSP (dall'inglese Travelling Salesman Problem), è un problema di teoria dei grafi, uno dei casi di studio tipici dell'informatica teorica e della teoria della complessità.

Il nome nasce dalla sua più tipica rappresentazione: data una rete di città, connesse tramite delle strade, trovare il percorso di minore lunghezza che un commesso viaggiatore deve seguire per visitare tutte le città una e una sola volta.

Espresso nei termini della teoria dei grafi è così formulato: dato un grafo completo pesato, trovare il ciclo hamiltoniano con peso minore. La rete di città può essere rappresentata come un grafo in cui le città sono i nodi, le strade gli archi e le distanze i pesi sugli archi.

Può essere dimostrato che specificare o meno il ritorno alla città di partenza non cambia la complessità computazionale del problema.

Il problema è di considerevole importanza pratica, al di là delle ovvie applicazioni nella logistica e nei trasporti. Un esempio classico è la costruzione di circuiti stampati, nella pianificazione del percorso del trapano per creare i fori nella piastra. Nelle applicazioni di foratura o di rifinitura automatica eseguite da robot, le "città" sono pezzi da rifinire o fori (anche di varie dimensioni) da praticare, e il "costo di viaggio" include i tempi morti (ad esempio il tempo che il robot impiega, se necessario, per cambiare la punta con cui lavora).

2.2.1 Complessita computazionale

Non esistono algoritmi efficienti per la risoluzione del TSP, l'unico metodo di risoluzione è rappresentato dall'enumerazione totale, ovvero nell'elaborazione di tutti i possibili cammini sul grafo per la successiva scelta di quello migliore. Tuttavia, la complessità dell'operazione la rende impraticabile per grafi di dimensioni comuni nei problemi reali: in un grafo di n nodi, bisognerà calcolare, nel caso peggiore in cui ogni nodo è connesso con tutti gli altri, n! (n fattoriale) possibili cammini, il che implica una complessità esponenziale (in base all'approssimazione di Stirling).

Il TSP rappresenta inoltre un esempio di problema di programmazione lineare intera nel quale risulta pressoché inattuabile ottenere valutazioni approssimate tramite la tecnica del rilassamento continuo poiché il problema di PL risultante si troverebbe ad avere un numero di vincoli che cresce in modo esponenziale con i nodi, rendendo così intrattabili le matrici associate ad esso. Per scopi pratici, il metodo della ricottura simulata (simulated annealing) ha effettivamente risolto il TSP.

Una rete di mille nodi, molto più comune di quanto si possa pensare, comincerebbe già a creare seri problemi computazionali.

2.2.2 NP-completezza

È stato dimostrato che TSP è un problema NP-difficile (più precisamente, è NP-completo per la classe di complessità FPNP; v. problema di funzione), e la versione decisionale del problema ("dati i pesi e un numero x, decidere se ci sia una soluzione migliore di x") è NP-completa.
Il problema rimane NP-difficile anche in molte sue versioni ridotte, come nel caso in cui le città siano in un piano con distanze euclidee. Inoltre, omettere la condizione di visitare una città "una e una sola volta" non rimuove la NP-difficoltà, poiché si nota facilmente che nel caso piano un cammino ottimale visiterebbe comunque le città una volta sola.

2.2.3 Algoritmi

Le strategie tradizionali di soluzione per i problemi NP-difficili sono:

- progettare algoritmi per trovare la soluzione esatta, ragionevolmente veloci solo per problemi con un numero di nodi relativamente basso

- progettare algoritmi euristici, cioè algoritmi che producono soluzioni probabilmente buone, ma impossibili da provare essere ottimali

- trovare un caso specifico del problema (sottoproblema) per il quale sia possibile o una soluzione esatta o un'euristica migliore.

Per condurre test sugli algoritmi TSP è stata sviluppata la libreria TSPLIB, che contiene istanze d'esempio del problema TSP e relative varianti (v. nella sezione voci correlate). Molti di questi esempi sono liste di città e schemi di circuiti stampati.

2.3 TSP per la cella robotizzata

Nel creare il percorso del robot si deve ottimizzare il costo complessivo dell'inserimento delle curve. Se consideriamo il punto di inserimento delle curve come dei nodi, il percorso che dovrebbe fare il robot consiste nel passaggio da tutti i nodi, minimizzando così il costo di movimento. In questa prospettiva quindi si nota una somiglianza tra questo problema e quello del commesso viaggiatore, con la sola differenza che, nel nostro caso specifico, per ogni punto di inserimento abbiamo due nodi diversi. Il TSP quindi, sarà vincolato da questo sdoppiamento, da prendere in considerazioni.

Il costo computazionale nel nostro caso é n! perché abbiamo ogni nodo che può essere collegato con tutti gli altri in relazione alla velocità di rotazione e di traslazione del Robot, perché utiliziamo tutto lo spazio e non c'é nessun vincolo in corrispondenza ai passaggi tra i nodi, il che rende difficile l'implementazione di un algoritmo che risolve il problema basato sul criterio TSP.

Capitolo 3
Algoritmi proposti

In questo capitolo andremo ad analizzare tutta la parte software che gestisce la cella, a partire dalla gestione dei DataBase degli scambiatori e degli ordini fino all'analisi degli algoritmi che costituiscono il cuore della cella.
L'ambiente di sviluppo con il quale é stato scritto l'intero programma di gestione della cella é MatLab, tale scelta e stata dettata dalla grande potenza e versatilità di questo software.

3.1 Strutture di dati

La struttura di dati principale che é stata usata durante il progetto é la matrice R, la quale contiene tutte le terne robot dirette e simmetriche. La matrice viene creata dalla funzione testaRobot (che vedremo dettagliatamente in seguito), a partire della matrice M che contiene le coordinate di tutti i tubi dello scambiatore, C matrice di descrizione curve e H matrice coordinate dei centri curve .

Nella tabella 2.1 si trovano tutte le strutture di dati usati dal software di gestione con una piccola discrizione, e sono utilizati in varie fasi dell'esecuzione del

DatiScamb{i}			**Descrizione**
.dati	.modello		nome del modello
	.tipo		"equilatero" o "non equilatero"
	.tubi		numero di tubi totale
	.I		interassi: $[25, 22.743]$ $[mm]$
	.h		distanza tra i ranghi $[mm]$
	.curv		numero di curve da 25 e $22.743mm$
	.rng		numero di ranghi
	.tpr		numero di tubi per rango
	.dtubo		diametro dei tubi $[mm]$
	.spalla		posizione spalla nel disegno CAD
	.hspallamax		altezza massima delle spalle $[mm]$
	.M		matrice di descrizione schema-tubi: M(i,:)=[x_{CAD}, y_{CAD}, Rango, Curva]
	.C		matrice di descrizione curve: C(i,:)=[Tubo1, Tubo2, Rango, Interasse]
	.H		matrice coordinate dei centri curve : H(i,:)=[x_{CAD}, y_{CAD}, Inclinazione°]
.cad	.name		nome del relativo file CAD
	.date		data inserimento file CAD nel database
	.bytes		dimensione del file CAD [byte]
.output{j}	.R		matrice delle terne Robot
	.pinza{k}	.NomePinza	nome della pinza
		.IDHardware	bit identificativi pinza
		.Interasse	interasse curve della pinza
		.DiamTubo	diametro dei tubi delle curve della pinza
		.dx	parametro pinza
		.dy	parametro pinza
		.Theta	parametro pinza
		.T_ct	matrice trasf. curva/tool
	.TRJ		matrice traiettoria ottima: TRJ(i,:)=[x, y, theta, curv, R, S]
	.s_tot		spazio percorso seguendo TRJ $[mm]$
	.theta_tot		rotazione totale pinza seguendo TRJ [°]
	.t_tot		tempo inserimento curve $[s]$

Tabella 3.1: Struttura del database DatiScamb.

3.2 Funzioni

3.2.1 Funzione principale (importScamb)

La funzione principale che crea il percorso é importScamb. Quest'ultima, effettua le seguenti operazioni: importazione-visualizzazione dello schema, caricamento pinze, scelta del file che contiene lo schema dello scambiatore, crea l'area grafica, chiama la funzioni testaRobot (che vedremo più avanti) che crea la matrice R con terne robot simmetrica per ogni curva, poi calcola il percorso ottimale e ristiuisce TRJ matrice con le coordinate terna Robot del percorso ed in fine traccia il disegno .

3.2.1.1 Funzione testaRobot

La funzione testaRobot crea la matrice R(1x9) contenente le coordinate della terna robot (R) e le coordinate della terna robot simmetrica rispetto alla curvetta (S) nel seguente ordine: R(i,:) = (R_x , R_y , R_theta , S_x , S_y , S_theta, numero Curva, theta_d, theta_s). Le prime 6 colonne di R descrivono il frame della testa robot associato alla curva diretta o a quella simmetrica. Le ultime 3 colonne, theta_d e theta_s descrivono il frame del baricentro della curvetta associato alla curva diretta o a quella simmetrica. Theta è compreso tra π e -π.
. Gli input che vengono utilizzati nella funzione sono i seguenti:

M: matrice contenente le coordinate (in mm) di tutti i tubi costituenti lo scambiatore, riferite ad un sistema di assi x-y ortogonale e centrato sul primo tubo inbasso a sinistra dello schema.

H: matrice contenente le coordinate (in mm) del baricentro delle curve rispetto allo stesso sistema di assi ortogonale x-y definito precedentemente.

C: matrice contenente il numero dei tubi collegati dalle curve; ad esempio, se la prima curvetta collega il tubo 1 con il tubo 6 nella matrice C, nella prima colonna vi sarà 1 e nella seconda colonna alla stessa riga vi sarà 6.

E' da tenere ben presente che la conoscenza del sistema di riferimento CAD è di fondamentale importanza per riportare le coordinate presenti nelle matrici in input nel sistema di riferimento robot. In realtà quello che noi sappiamo è solamente l'orientazione di tale sistema di riferimento (rif. CAD) ma poichè in questo algoritmo andremo a fare esclusivamente differenze tra coordinate, conoscere l'origine del frame non ci interessa.

Nel nostro caso abbiamo che i due sistemi di riferimento sono così disposti:

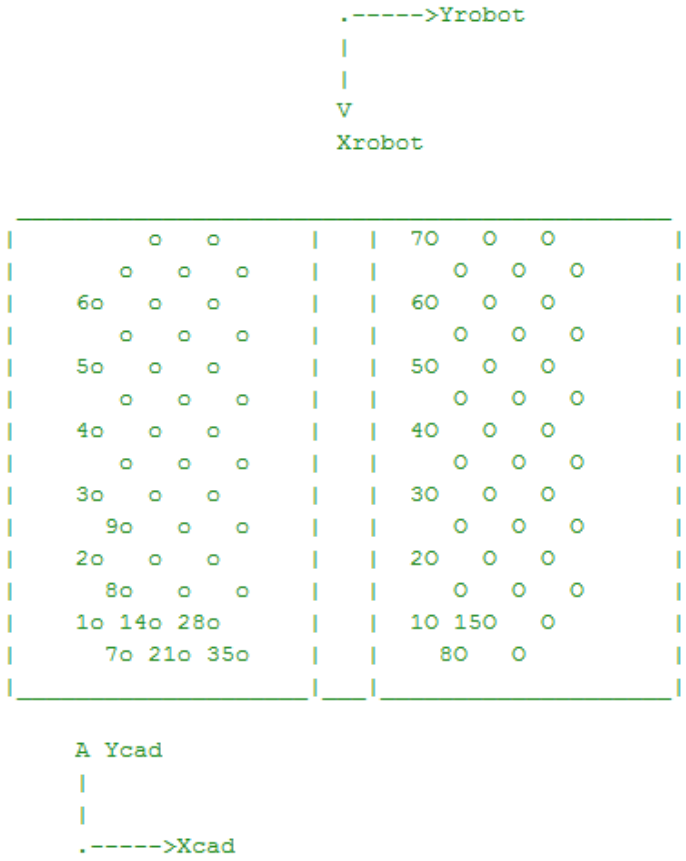


Figura 3.1 i due sistemi di riferimento

Si tenga quindi presente che nella matrice R non sono presenti le vere coordinate delle curvette (diritte e simmetriche), ma sono presenti le coordinate definite in un sistema di riferimento con orientazione identica a quella della terna world(robot).
L'angolo theta non è più riferito al sistema di coordinate dello scambiatore bensì alla terna robot.

3.2.2 Funzioni per disegnare il percorso.

La funzione princepale, disPath, disegna il percorso a partire dalla matrice ottima TRJ.
INPUT:
- handles = necessari per aggiornare GUI.
- TRJ = matrice ottima delle terne dato essenziale da inserire per mantenere la corrispondenza con la matrice C nel caso lo schema sia non equilatero.
- frame = >0 se si vuole visualizzare le terne curvette (=0 se no).
- style = disegnare simbolo agl'apici degl'assi (robot e curvette se abilitate).
- offset: serve quando si deve disegnare nella rappresentazione grafica il secondo scambiatore, altrimenti viene sovrapposto al primo.

All'interno della funzione disPath abbiamo altre funzioni di grafica che sono: disframe, disCurva, disTraiettoria .

disframe visualizza una terna di riferimento e ha come input :
- T : matrice di rototraslazione dal sistema locale a quello che fa da riferiento per le immagini (assoluto).
- L : lunghezza dei vettori da disegnare per visualizzare gli assi.
- stile : tipo di punto da visualizzare all'estremità degli assi.

disCurva disegna una curva ed ha come input :
- frame disegna le terne di riferimento delle curvette se >0.
- style disegna un punto o croce o stella all'estremità degl'assi.
- index permette di plottare una qualunque sequenza di curve da quella originale a quella ottima data dallapianificazione.

disTraiettoria disengna la traiettoria a partire dalle coordinate (x,y) dirette e simmetriche ed ha come input anche offset che serve quando si deve disegnare il secondo scambiatore altrimenti viene sovrapposto al primo.

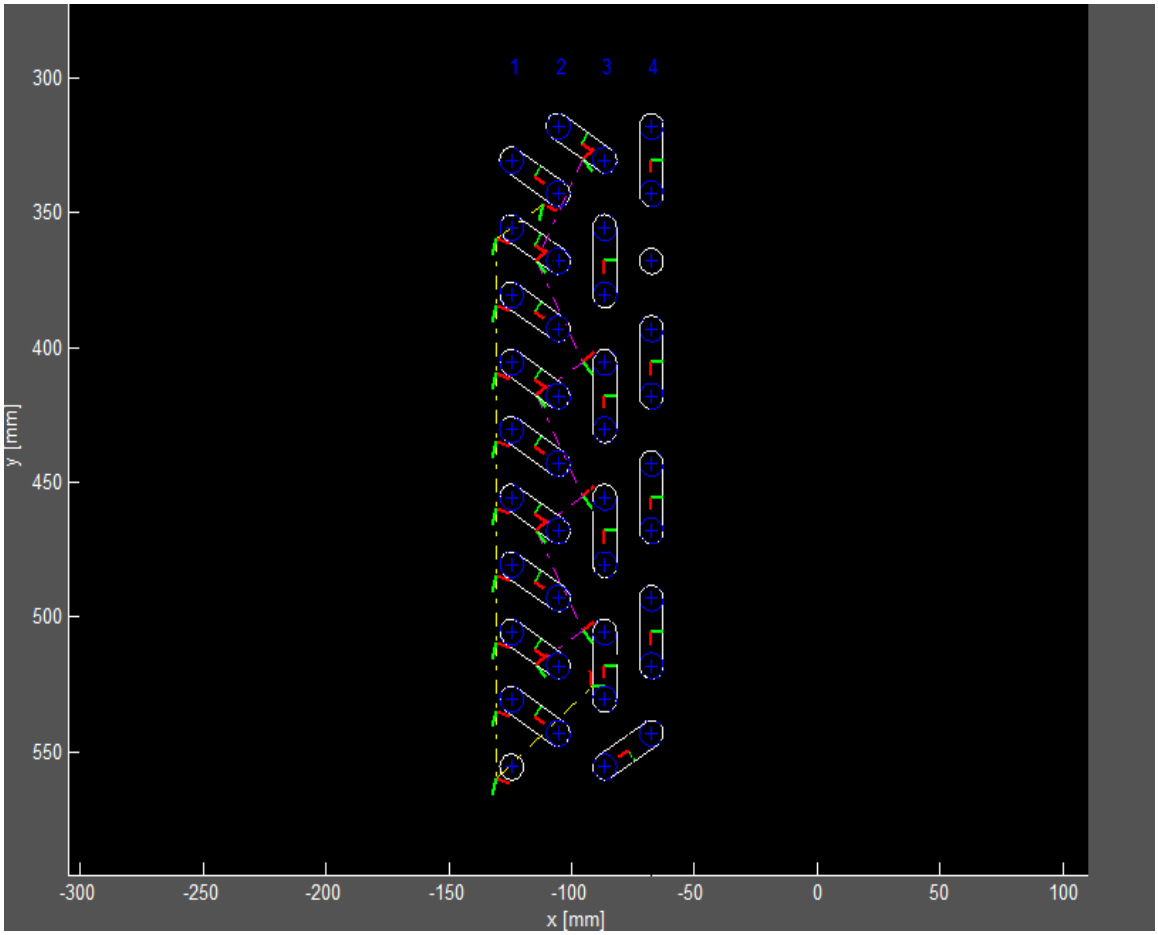


Figura 3.2 disegno di una traiettoria

3.2.3 minPath

E' la funzione che calcola il percorso minimo ed é basata su un algoritmo che usa la procedura di Djikstra. Quest'ultimo, modificato per il nostro caso, dato che il problema é quello di passare per tutti i nodi, abbiamo utilizato il criterio su una matrice R che contiene le coordinate delle curvette (sia in forma diritta che simmetrica) e le coordinate dei relativi baricentri rispetto alla terna robot. In primo luogo, abbiamo oridinato la matrice in modo crescente in relazione alla seconda colona (cordinate y della terna diretta, che ordina automaticamente quelle simetriche). Nel caso in cui ci si trova di fronte ad una situazione d'ugualanza è neccessario fare riferimento ai dati della prima colonna, in modalità decrescente, al fine di ricavare il punto di partenza, che quindi sarà quello più alto a senistra.
In secondo luogo, conclusa la fase del sort, si procede con la creazione della matrice RS. Quest'ultima è la matrice che contiene tutte le terne (x,y,theta) dirette e simmetriche , il numero di curva e 1 se é diretta altrimenti 0, ed in fine theta baricentro.

In terzo luogo, si procede con il calcolo della matrice J, ovvero la matrice di costo, creata dalla matrice RS calcolando le due matrici [xy] e [gg] delle distanze (cartesiane e angolari). Per ultimare il calcolo della matrice J si procede con il calcolo della matrice di distanza temporale eseguendo la seguente operazione: divisione delle due matrici sopra citate, per le due velocità corrispondenti v e w (traslazione e angolare). arrivati a questo punto, avremo matrice J come valore massimo tra le due matrice. In ultima analisi, riportiamo a 0 le distanze corrispondenti alla stessa curvetta e quella ad essa simmetrica.

```
R = scamb.output{nout}.R;
R = sortrows(R,[2 1]);
% numero delle terne
N = size(R,1);
% creazione della matrice RS
YR = [ R(:,1:3) R(:,7) ones(N,1) zeros(N,1) R(:,8)];
YS = [ R(:,4:6) R(:,7) 2*ones(N,1) zeros(N,1) R(:,9)];
RS = [ YR; YS ];
% questa è la matrice che contiene tutte le terne [R_x , R_y , R_theta ,numero_curva_diritta,   1, 0, theta_bar_diritto   ;
%                                                  S_x , S_y , S_theta ,numero_curva_simmetrica 2, 0, theta_bar_simmetrico]
% matrice delle coordinate cartesiane ed angolari delle terne in [mm] e [°]
%replica la prima colonna con 2*N righe per altre 2*N colonne
xx = repmat(RS(:,1),1,size(RS,1));
yy = repmat(RS(:,2),1,size(RS,1));
tt = repmat(RS(:,3),1,size(RS,1));
% matrici delle distanze tra le terne i e j
xy = sqrt( (xx-xx').^2 + (yy-yy').^2 );        %[m] distanza cartesiana
gg = (pi/180)*min(abs(tt-tt'),abs(360+tt'-tt));     %[rad] distanza angolare (ultimo giunto)
xy=xy/v;%matrice distanze temporali
gg=gg/w;%matrice angoli/velocita
J=max(xy,gg);
% escludo le terne abbinate alla stessa curvetta se
% entrambe le terne sono corrette, infatti in caso contrario la matrice
% xy ha già posto = 0 nella posizione (i,j) e (j,i)
for i = 1:(length(RS(:,1))/2)
    tipp = RS(i,4);
    ind=find(RS(:,4)==tipp);
    if length(ind)>1 % entrambe le orientazioni sono corrette
        J(ind(1),ind(2)) = 0;
        J(ind(2),ind(1)) = 0;
    end
end
```

Figura 3.3 procedura per la creazione della matrice J

Una volta creata la matrice J, si inserisce il punto di partenza, che corrisponde alla prima riga della matrice RS.
A questo punto inizia il ciclo principale dell' algoritmo, ovvero é un "For" che va da 1 fino a N-1 curve (poiché abbiamo già inserito il primo).
All'interno del cicolo, il valore delle due colonne della matrice J che corrispondono alla prima curva inserita, vengono ridotte a 0 perché non vengano più conteggiate. Dopo di che segue l'aggiornamento dei parametri step e nodi che sono i parametri principali nella scelta del percorso e che ci permettono di avere diversi percorsi,si cerca i primi nodi vicini dopo se crea per ciascuno un percorso cercando il vicino per step volte.

Il parametro "alg" viene usato per la scelta di un algoritmo tra i tre proposti, viene dichiarato in "importScamb" e dopo passato come parametro a minPath.
i parametri "step" e "nodi" vengono aggiornati dentro il ciclo quando diventano maggiore al numero di curve rimanente, cosi si evitano operazione inutile.
la funzione "finMinStep" che fa tornare un matrice "indici" e il tempo complessivo, alla fine viene scelto l'elemento oppure l'indici della matrice J del minimo costo , inserendo il nodo nella matice TRJ, sempre con l eliminazione dei nodi della curva scelta (diretto e simmetrico) si comincia un altro giro.

Nella Figura 3.4 si vede il corpo del ciclo principale nel minPath inseguito annalisiamo ogni algoritmo in dettaglio ma prima vedremo la funzione "findMinStep" che é il cuore del programma

```
for i=1:N-1
    %eliminare le due colonne della matrice j
    J(:,indice)=0;
    if indice<=N
        J(:,indice+N)=0;
    else
        J(:,indice-N)=0;
    end
    indice_old = indice;
    if N-i>nodi   % correzione numero nodi
        nn = nodi;
    else
        nn = N-i;
    end
    if N-i>step   % correzione numero step
        ss = step;
    else
        ss = N-i;
    end
    [indici,tempo] = findMinStep(J,indice,nn,ss,alg);
    indice = indici(1);
    % correzione indice
    if indice<=N
        indice2 = indice+N;
    else
        indice2 = indice-N;
    end
    if J(indice_old,indice)>J(indice_old,indice2)
        indice = indice2;
    end
    t_tot = t_tot + J(indice_old,indice);
    TRJ = vertcat(TRJ, RS(indice,1:6));
end
```

Figura 3.4 il ciclo princpale della funzione minPath

3.2.3.1 findMinStep

E' una funzione ricorsiva che restituisci una matrice con gli indici e costi di tutti i percorsi trovati, la ricorsione ha una profondità di step-1, nella figura seguente vediamo in dettaglio la funzione

```
function [indnext,tempo] = findMinStep(J,indatt,nodi,step,alg)

% trova i nodi da provare
switch alg
    case 1 % minima distanza (più vincio)
        I = minriga(J(indatt,:),J(indatt,:),nodi);
    case 2 % più esterno allo schema (reciproco della somma delle distanze da tutte le curve)
        I = minriga(J(indatt,:),1./sum(J,2)',nodi);
    case 3 % pesato (somma di distanza minore e distanza media, calcolata come media delle differenze tra
           % massima somma delle distanze e somma delle distanze della curva considerata)
        I = minriga(J(indatt,:),.05*J(indatt,:)+(max(sum(J,2))-sum(J,2)')/sum(find(J(1,:)>0)),nodi);
end
if step == 1
    % all'ultimo step, restituisce il nodo più vicino
    indnext = I(1,1);
    tempo = I(2,1);
else % altrimenti, cerca i nodi più vicini e prosegue
    N = size(J,1)/2;
    for j=1:nodi
        Jnew = J;
        Jnew(:,I(1,j)) = 0;
        if I(1,j)<=N
            Jnew(:,I(1,j)+N)=0;
        else
            Jnew(:,I(1,j)-N)=0;
        end
        % proseguo con uno step in meno
        [ind{j},t(j)] = findMinStep(Jnew,I(1,j),nodi,step-1,alg);
    end
    jmin = find(t+I(2,:)==min(t+I(2,:)),1);
    indnext = [I(1,jmin) ind{jmin}];
    tempo = t(jmin) + I(2,jmin);

end
```

Figura 3.5 findMinstep

"minriga" la funzioni che cerca e restituisci i nodi pi vicini, in pratica esegue due ordinamenti il primo basato sulle distanze temporale una volta scelto i nodi si effettua un altro ordinamento secondo il criterio prescelto per l'algoritmo tramite il parametro "alg".

```
function ind = minriga(A,B,n)

Ainf = A;
Ainf(A==0) = Inf;

[~,iA] = sort(Ainf);

indA = iA(1:n);

[~,iB] = sort(B(indA));

ind = [indA(iB); B(indA(iB))];
```

figura 3.6 minriga

Nei prossimi capitoli andremo ad analisare i diversi algoritmi proposti e definiamo esplicitamente le due matrice riga che vengono passati alla funzione minriga secondo il criterio scelto.

3.2.3.2 Algoritmo basato su distanze

Come e stato accennato prima l'algoritmo si basa su il criterio base, che significa che per ogni posizione andremo a cercare il nodo pi vicino cosi via fino che si passa da tutti i nodi, abbiamo anche introdotto i due parametri nodi e stap solo che un questo caso non si ottenga nessuno effetto visto che per ogni step si cerca sempre il più vicino e anche per la struttura geometrica regolare su cui viene eseguito il programma, comunque vedremo questi limiti nel quarto capitolo.

La funzione minriga in questo caso avrà come parametri dei numero nodi aggiornato e due righe che contengono le distanze dal nodi attuale:

```
I = minriga(J(indatt,:),J(indatt,:),nodi);
```

3.2.3.3 Algoritma Curva-esterna

Diversamente dal primo algoritmo, alla funzione minriga viene passato una riga che contiene il reciproco della soma di tutte le distanze per ogni nodo, cosi si restituisci una matrice con i nodi vicini ma più esterne. un nodi ed esterno ha una somma delle distanze maggiore, abbiamo passato il reciproco solo per non creare un altra funzione ed utilizzare la funzione minriga.

l'algoritmo é più compatto, evita (al contrario del primo algoritmo) il problema di lasciare curve non inserite e per conseguenza fare giri in più o percorsi non regolare, pero ci constata certi salti quando si aumenta il numero dei nodi vicini da scegliere, perché cerca di andare al vicino ma il più esterno.Vediamo come venga dichiarata la funzione minriga:

```
I = minriga(J(indatt,:),1./sum(J,2)',nodi);
```

3.2.3.4 Algoritmo pesato

Ed un algoritmo che usa i due criteri proposti prima in modo implicito, andando a pesare i due matrice di costo utilizzate prima che si realizza come la somma di distanze minore e distanze media, calcolata come media delle differenze tra massima somma delle distanze e somma delle distanze della curva considerata . Cosi evitiamo il salto e avremo un percorso compatto visto che la regoralità geometrica non avrà un effetto in questo caso. La dichiarazione della minriga e la seguente :

```
I=minriga(J(indatt,:),.05*J(indatt,:)+(max(sum(J,2))-sum(J,2)')/sum(find(J(1,:)>0)),nodi);
```

Nel capitolo succesivo vedremo in dettaglio tutti i problemi e come sono risolti da varie algoritmi.

Capitolo 4

Risultati delle simulazioni

In questo capitolo riportiamo i risultati in tabelle dei test fatti su diversi schemi, e mostreremo anche la differenza tra gli tre algoritmi proposti utilizzando dei grafi fatti su una schema campione.

4.1 Test,simulazione e analisi

Vedremo alcuni grafici dei varie test su una schema che andremo ad analisare evidenziando i principali punti di differenza tra i test, i diversi algoritmi scelti e i limiti di ciascun algoritmo e come vengono superati.

Per tutti i test abbiamo stabilito un numero di nodi uguale a 3 con un unico step.

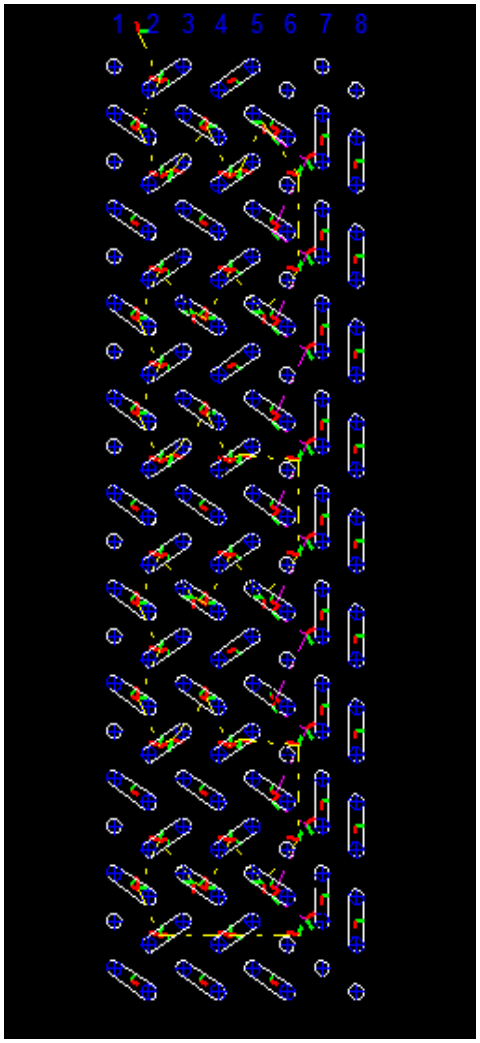


Figura 4.1 Percorso secondo l'algoritmo di distanza

Come si vede nella figura 4.1 e più precisamente nella figura 4.2 che il per-corso non é totalmente regolare facendo un zig-zag dovuto al fatto che inserici prima la curva di sotto puoi torna a quella sopra, il che aumenta il costo del movimento.

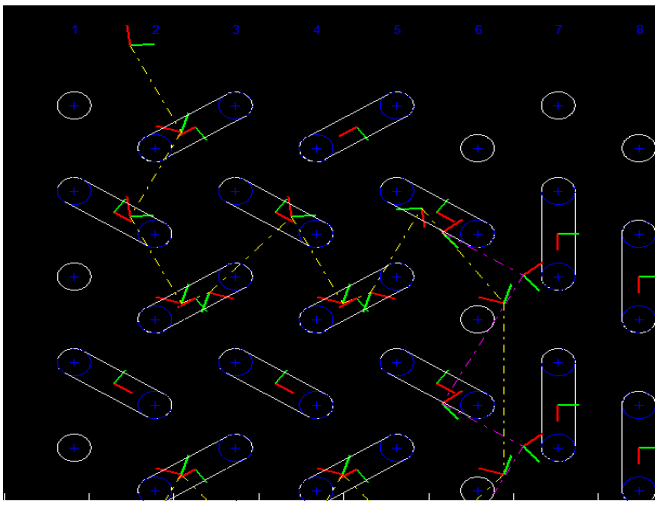


Figura 4.2 Il problema del'algoritmo di distanza

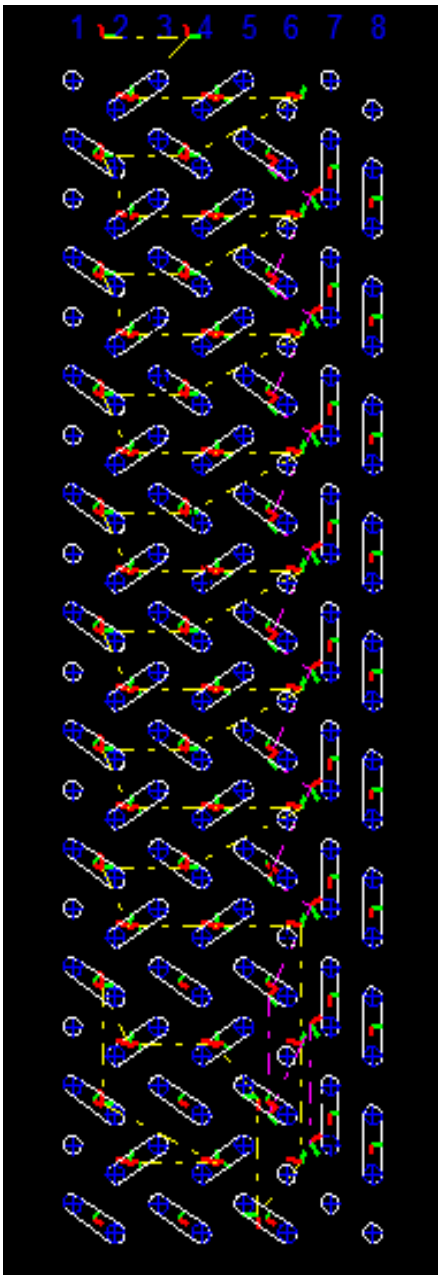


Figura 4.3 Percorso secondo l'algoritmo di curva esterna

Come si vede nella figura 4.2 e più precisamente nella figura 4.3 che il percorso é più regolare pero nel basso dello schema si constata un salto che produce un giro in più si vedi nella figura 4.4, perché praticamene in tale posizione visto che abbiamo stabilito numero dei nodi 3, tra quelli tre scelti i più esterno era l'ultimo per quello ha fato un tale salto che ha costato tanto visto il percorso fato dopo.

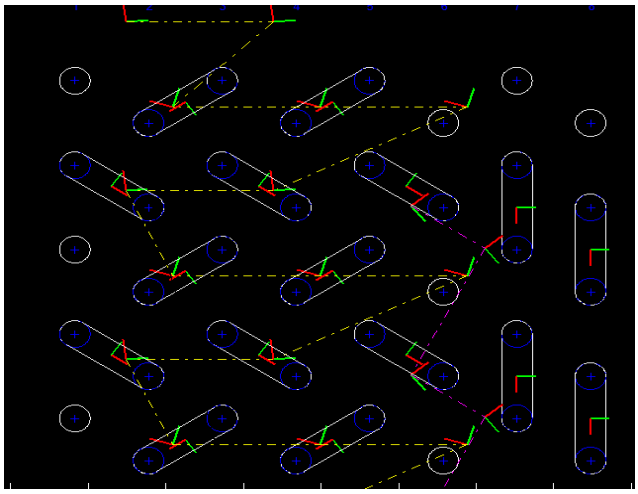


Figura 4.4 Percorso regolare al contrario del primo algoritmo

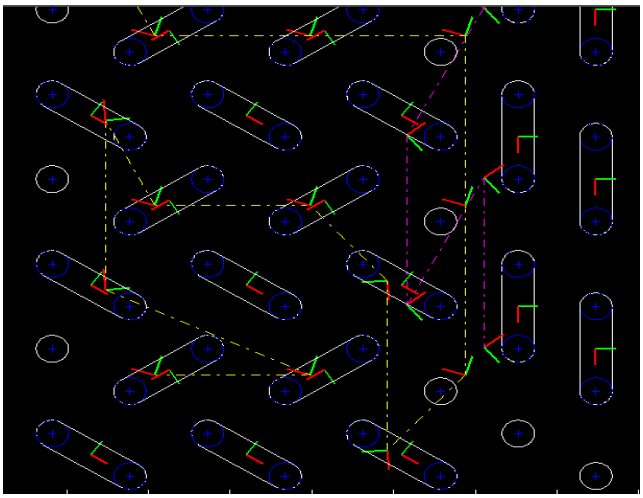


Figura 4.5 Salto nella parte bassa dello schema

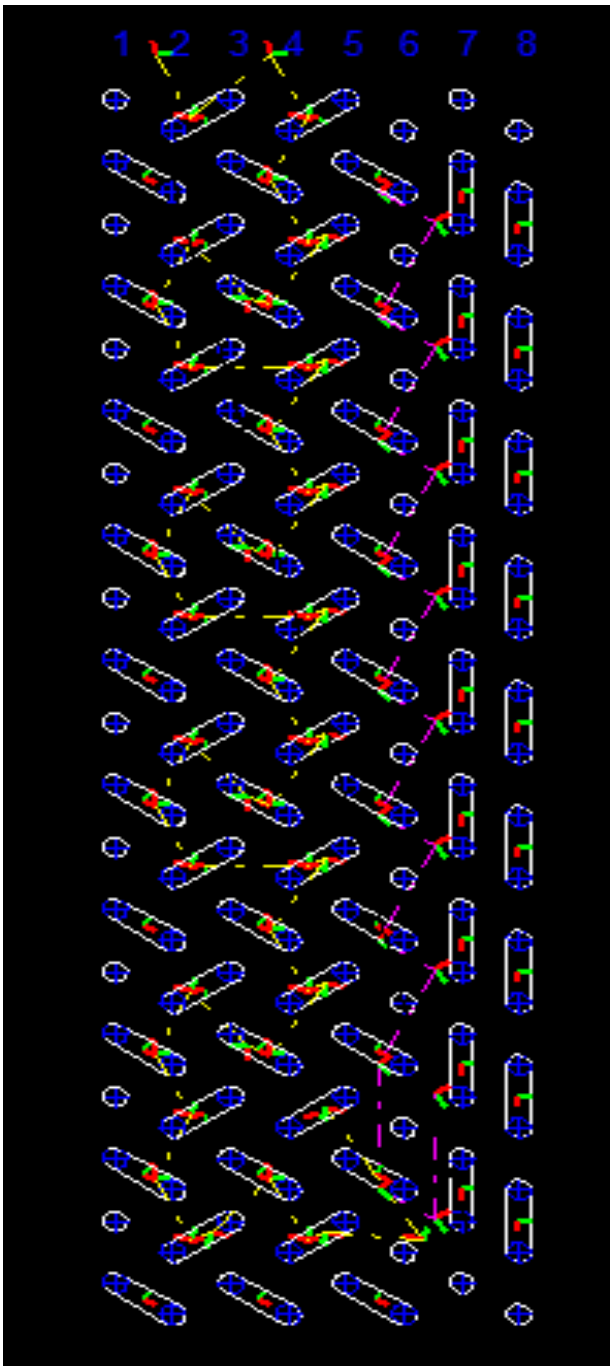


figura4.6 Percorso secondo l'algoritmo pesato

Nella figura 4.6 abbiamo il percorso usando l'ultimo algoritmo proposto come si vede ha superato i limiti incontrati dai due primi algoritmi,quello é dovuto al fatto che usa una procedura pesata tra i due criteri, come e mostrato nelle figure 4.7 e 4.8

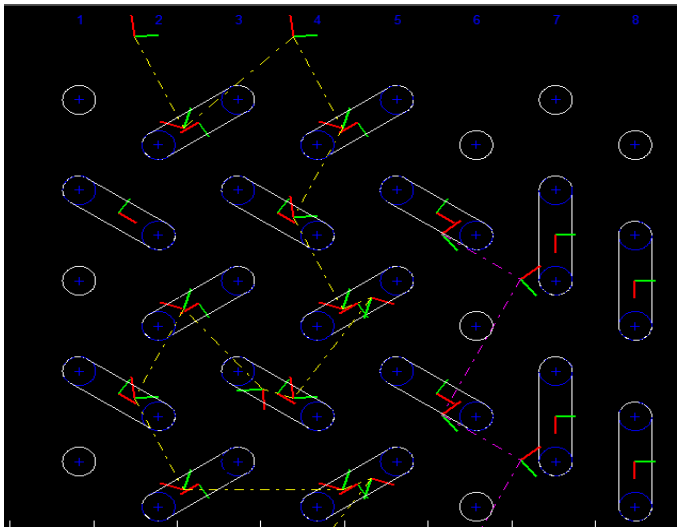
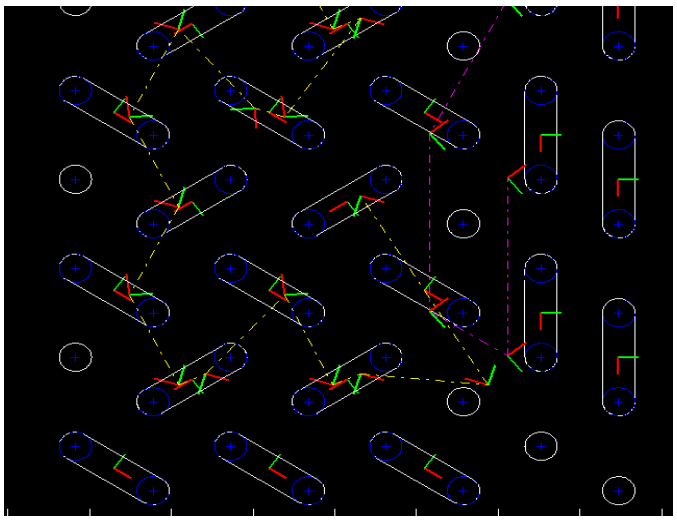


Figura 4.7 Percorso non regalare ma più efficiente



Fihura 4.8 Assenza del salto diversamente dal secondo algoritmo

Da questo test sembra che l'algoritmo pesato é il più ottimo, il che non é sempre vero, come andremo a vedere dopo, che al variare del tipo dello schema e anche delle variabile (nodi,step,velocità) cambia l'efficenza di ciascun algoritmo.

4.2 Tabelle riassuntivi dei risultati

In questo paragrafo vedremo tutti i risultati fatti su varie schemi variando i parametri "nodi", "step" e le due velocità.

I test sono fatti come seguito, scegliando la velocita di traslazione e angolare v e w dopo variando numero nodi e step per ciascun algoritmo.

Nei due primi tabelle 4.1 e 4.2 abbiamo fissato le velocità in 100 mm/s e 100 rad/s , per le altre due tabelle 4.3 e 4.4 diminuiamo la velocità angolare a 10 rad/s, mentre i numeri dei nodi e step varia per ciascun algoritmo e schema come é mostrato nelle tabelle dove viene riportato per ogni scelta il tempo totale impiegato con il rapporto percentuale rispetto al caso base.

Da precisare anche che tutti i tempi calcolati sono multiplicati per 10^4 questo per semplificare il calcolo del raportto percentuale

	Alg .Distanza (s)						Alg.Esterna (s)						Alg.Pesato (s)					
schemi	1/1		1/3		1/5		3/1		3/3		3/5		3/1		3/3		3/5	
I	64.169	100	64.169	100	64.169	100	89.037	139	79.170	123	74.860	117	62.887	98	52.499	82	58.084	91
II	129.445	100	129.445	100	129.445	100	195.180	151	190.656	147	204.907	158	131.099	101	122.201	94	137.076	106
	54.100	100	54.100	100	54.100	100	59.551	110	110.886	205	148.517	275	54.100	100	57.622	107	67.622	125
III	25.402	100	25.402	100	25.402	100	25.402	100	41.970	165	42.457	167	25.402	100	29.757	117	32.920	130
	23.638	100	23.638	100	23.638	100	26.763	113	39.736	168	39.736	168	24.435	103	28.693	121	45.187	191
IV	33.471	100	33.471	100	33.471	100	44.059	132	59.529	178	59.529	178	41.659	124	38.035	114	55.259	165
V	33.471	100	33.471	100	33.471	100	44.059	132	59.529	178	59.529	178	41.659	124	38.035	114	55.259	165
VI	69.436	100	69.436	100	69.436	100	52.049	75	75.318	108	77.059	111	52.049	75	67.717	98	77.646	112
	20000	100	2000	100	20000	100	20000	100	35.000	175	35.000	175	20000	100	35.000	175	35.000	175
VII	128.176	100	128.176	100	128.176	100	158.099	123	177.020	138	174.056	136	150.132	117	196.910	154	139.790	109
VIII	34.582	100	34.582	100	34.582	100	33.470	97	50.936	147	45.916	133	34.582	100	42.538	123	41.998	121
IX	40.456	100	40.456	100	40.456	100	39.495	98	51.026	126	75.565	187	39.495	98	40.456	100	70.655	175
X	91.633	100	91.633	100	91.633	100	93.516	102	123.178	134	159.290	174	91.633	100	91.633	100	98.024	107
	125.308	100	125.308	100	125.308	100	137.077	109	147.627	118	219.000	175	141.036	113	125.633	100	121.774	97
XI	36.952	100	36.952	100	36.952	100	49.587	134	59.551	161	76.327	207	47.187	128	45.493	123	61.220	166
XII	31.155	100	31.155	100	31.155	100	41.596	134	53.182	171	58.698	188	41.596	134	52.505	169	54.416	175
XIII	12.500	100	12.500	100	12.500	100	12.500	100	16.291	130	16.291	130	12.500	100	17.725	142	17.725	142
XIV	44.104	100	44.104	100	44.104	100	54.676	124	53.926	122	64.881	147	40.665	92	52.671	119	56.197	127
XV	33.800	100	33.800	100	33.800	100	51.292	152	48.215	143	60.041	178	39.000	115	43.900	130	48.648	144
	22.792	100	22.792	100	22.792	100	24.718	108	22.792	100	22.792	100	24.718	108	27.242	120	27.242	120
XVI	0	100	0	100	0	100	0	100	0	100	0	100	0	100	0	100	0	100
	53.577	100	53.577	100	53.577	100	60.699	113	67.424	126	71.596	134	58.656	109	46.724	87	77.724	145
XVII	398.140	100	398.140	100	398.140	100	341.505	86	350.153	88	384.603	97	293.229	74	454.485	114	511.232	128
	153.880	100	153.880	100	153.880	100	160.137	104	233.608	152	328.433	213	153.880	100	154.964	101	173.486	113
XVIII	208.450	100	208.450	100	208.450	100	212.184	102	232.080	111	268.941	129	208.450	100	201.699	131	198.499	129
XIX	79.052	100	79.052	100	79.052	100	79.052	100	112.679	143	213.518	270	79.052	100	79.052	100	99.449	126
	213.945	100	213.945	100	213.945	100	162.313	76	187.698	88	208.884	98	163.291	76	219.934	103	163.255	76
								112		**139**		**160**		**103**		**116**		**132**

Tabella 4.1 Tabella riassuntiva dei test su vari schemi con velocità v=100m/s e w=100rad/s

	Alg .Distanza (s)						Alg.Esterna (s)						Alg.Pesato (s)					
schemi	1_1		1_3		1_5		6_1		6_3		6_5		6_1		6_3		6_5	
I	64.169	100	64.169	100	64.169	100	86.937	126	80.077	116	81.787	118	66.237	96	61.700	89	76.573	111
	129.445	100	129.445	100	129.445	100	235.345	147	203.866	127	271.232	169	140.624	88	130.766	82	159.202	99
II	54.100	100	54.100	100	54.100	100	59.551	110	150.601	278	152.844	283	54.100	100	57.622	107	69.646	129
	25.402	100	25.402	100	25.402	100	27.409	100	44.446	162	54.869	200	27.409	100	54.869	200	57.302	209
III	23.638	100	23.638	100	23.638	100	24.435	103	45.924	194	50.432	213	24.435	103	37.706	160	52.254	221
IV	33.471	100	33.471	100	33.471	100	40.670	104	74.910	192	77.833	199	41.659	107	42.764	110	65.287	167
V	33.471	100	33.471	100	33.471	100	40.670	104	74.910	192	77.833	199	41.659	107	42.764	110	65.287	167
	69.436	100	69.436	100	69.436	100	55.743	80	87.606	126	103.322	149	52.049	75	67.717	98	87.690	126
VI	20000	100	20000	100	20000	100	2	100	35.000	175	35.000	175	20000	100	35.000	175	35.000	175
VII	128.176	100	128.176	100	128.176	100	169.776	130	225.108	172	256.053	196	150.132	115	196.910	151	134.621	103
VIII	34.582	100	34.582	100	34.582	100	36.147	105	42.097	122	46.031	133	34.582	100	43.653	126	39.479	114
IX	40.456	100	40.456	100	40.456	100	39.495	109	69.665	193	68.604	190	35.835	99	68.025	188	71.221	197
	91.633	100	91.633	100	91.633	100	93.516	102	154.037	168	273.638	299	91.633	100	91.633	100	91.633	100
X	125.308	100	125.308	100	125.308	100	149.473	76	191.935	97	344.235	175	141.036	72	130.643	66	129.100	65
XI	36.952	100	36.952	100	36.952	100	55.071	125	91.504	208	84.872	193	47.187	108	44.763	102	76.607	175
VII	31.155	100	31.155	100	31.155	100	50.500	124	51.502	127	80.851	199	45.103	111	64.241	158	59.491	146
XIII	12.500	100	12.500	100	12.500	100	12.500	100	16.291	130	16.291	130	12.500	100	17.725	142	17.725	142
XIV	44.104	100	44.104	100	44.104	100	74.971	136	64.542	117	84.977	155	48.029	87	60.082	109	58.559	107
	33.800	100	33.800	100	33.800	100	51.292	152	64.472	191	66.714	197	39.000	115	55.094	163	59.502	176
XV	22.792	100	22.792	100	22.792	100	24.264	106	22.792	100	22.792	100	24.264	106	31.062	136	31.062	136
	0	100	0	100	0	100	0	100	0	100	0	100	0	100	0	100	0	100
XVI	53.577	100	53.577	100	53.577	100	51.637	85	69.473	114	86.813	143	58.275	96	77.300	127	84.890	139
	398.140	100	398.140	100	398.140	100	341.505	83	355.396	87	531.604	130	296.270	72	454.485	111	514.537	126
XVII	153.880	100	153.880	100	153.880	100	158.383	102	296.936	191	412.942	266	153.880	99	154.964	100	194.139	125
XVIII	208.450	100	208.450	100	208.450	100	294.460	141	293.668	141	381.446	183	208.450	100	288.884	138	224.919	108
	79.052	100	79.052	100	79.052	100	79.052	100	176.874	224	254.884	322	79.052	100	79.052	100	105.116	133
XIX	213.945	100	213.945	100	213.945	100	142.649	69	196.976	96	279.459	136	163.291	80	219.934	107	184.708	90
								108		**153**		**183**		**98**		**124**		**137**

Tabella 4.2 Tabella riassuntiva dei test su vari schemi con velocità v=100m/s e w=100rad/s

	Alg .Distanza (s)						Alg.Esterna (s)						Alg.Pesato (s)					
schemi	1_1		1_3		1_5		3_1		3_3		3_5		3_1		3_3		3_5	
I	69.158	100	69.158	100	69.158	100	89.037	129	79.170	114	74.860	108	62.887	91	52.499	76	58.084	84
	160.346	100	160.346	100	160.346	100	195.180	122	190.656	119	204.907	128	131.099	82	122.201	76	137.076	85
II	54.100	100	54.100	100	54.100	100	59.551	110	110.886	205	148.517	275	54.100	100	57.622	106	67.622	125
	27.409	100	27.409	100	27.409	100	25.402	93	41.970	178	42.457	155	25.402	93	29.757	126	32.920	139
III	23.638	100	23.638	100	23.638	100	26.763	113	39.736	168	39.736	168	24.435	103	28.693	121	45.187	191
IV	39.025	100	39.025	100	39.025	100	44.059	113	59.529	153	59.529	153	41.659	107	38.035	97	55.259	142
V	39.025	100	39.025	100	39.025	100	44.059	113	59.529	153	59.529	153	41.659	107	38.035	97	55.259	142
	69.436	100	69.436	100	69.436	100	52.049	75	75.318	108	77.059	111	52.049	75	67.717	98	77.646	112
VI	20000	100	20000	100	20000	100	20000	100	35.000	175	35.000	175	20000	100	35.000	175	35.000	175
VII	130.703	100	130.703	100	130.703	100	33.470	26	50.936	39	45.916	35	34.582	26	42.538	33	41.998	32
VIII	34.582	100	34.582	100	34.582	100	33.470	97	50.936	147	45.916	133	34.582	100	42.538	123	41.998	121
IX	36.132	100	36.132	100	36.132	100	39.495	109	51.026	141	75.565	2.091	39.495	109	40.456	112	70.655	196
	91.633	100	91.633	100	91.633	100	93.516	102	123.178	134	159.290	174	91.633	100	91.633	100	98.024	107
X	197.209	100	197.209	100	197.209	100	137.077	70	147.627	75	219.000	111	141.036	72	125.633	64	121.774	62
XI	43.895	100	43.895	100	43.895	100	49.587	113	59.551	136	76.327	174	47.187	108	45.493	104	61.220	139
VII	40.695	100	40.695	100	40.695	100	41.596	102	53.182	131	58.698	144	41.596	102	52.505	129	54.416	134
XIII	12.500	100	12.500	100	12.500	100	12.500	10	16.291	130	16.291	130	12.500	100	17.725	142	17.725	142
XIV	54.930	100	54.930	100	54.930	100	54.676	100	53.926	98	64.881	118	40.665	74	52.671	96	56.197	102
	33.800	100	33.800	100	33.800	100	51.292	152	48.215	143	60.041	178	39.000	115	43.900	130	48.648	144
XV	22.792	100	22.792	100	22.792	100	24.718	108	22.792	100	22.792	100	24.718	108	27.242	120	27.242	120
	0	100	0	100	0	100	0	100	0	100	0	100	0	100	0	100	0	100
XVI	60.877	100	60.877	100	60.877	100	60.699	100	67.424	111	71.596	118	58.656	96	46.724	77	77.724	128
	409.575	100	409.575	100	409.575	100	341.505	83	350.153	85	384.603	94	293.229	72	454.485	111	511.232	125
XVII	155.485	100	155.485	100	155.485	100	160.137	103	233.608	150	328.433	211	153.880	99	154.964	100	173.486	112
XVIII	208.871	100	208.871	100	208.871	100	212.184	102	232.080	111	268.941	129	208.450	100	201.699	97	198.499	95
	79.052	100	79.052	100	79.052	100	79.052	100	112.679	143	213.518	270	79.052	100	79.052	100	99.449	126
XIX	205.315	100	205.315	100	205.315	100	162.313	79	187.698	91	208.884	101	163.291	80	219.934	107	163.255	207
								97		**127**		**216**		**93**		**104**		**125**

Tabella 4.3 Tabella riassuntiva dei test su vari schemi con velocità v=100m/s e w=10rad/s

	Alg .Distanza (s)						Alg.Esterna (s)						Alg.Pesato (s)					
schemi	1_1		1_3		1_5		3_1		3_3		3_5		3_1		3_3		3_5	
I	69.158	100	69.158	100	69.158	100	89.037	129	79.170	114	74.860	108	62.887	91	52.499	76	58.084	84
	160.346	100	160.346	100	160.346	100	195.180	122	190.656	119	204.907	128	131.099	82	122.201	76	137.076	85
II	54.100	100	54.100	100	54.100	100	59.551	110	110.886	205	148.517	275	54.100	100	57.622	106	67.622	125
	27.409	100	27.409	100	27.409	100	25.402	93	41.970	178	42.457	155	25.402	93	29.757	126	32.920	139
III	23.638	100	23.638	100	23.638	100	26.763	113	39.736	168	39.736	168	24.435	103	28.693	121	45.187	191
IV	39.025	100	39.025	100	39.025	100	44.059	113	59.529	153	59.529	153	41.659	107	38.035	97	55.259	142
V	39.025	100	39.025	100	39.025	100	44.059	113	59.529	153	59.529	153	41.659	107	38.035	97	55.259	142
	69.436	100	69.436	100	69.436	100	52.049	75	75.318	108	77.059	111	52.049	75	67.717	98	77.646	112
VI	20000	100	20000	100	20000	100	20000	100	35.000	175	35.000	175	20000	100	35.000	175	35.000	175
VII	130.703	100	130.703	100	130.703	100	33.470	26	50.936	39	45.916	35	34.582	26	42.538	33	41.998	32
VIII	34.582	100	34.582	100	34.582	100	33.470	97	50.936	147	45.916	133	34.582	100	42.538	123	41.998	121
IX	36.132	100	36.132	100	36.132	100	39.495	109	51.026	141	75.565	2.091	39.495	109	40.456	112	70.655	196
	91.633	100	91.633	100	91.633	100	93.516	102	123.178	134	159.290	174	91.633	100	91.633	100	98.024	107
X	197.209	100	197.209	100	197.209	100	137.077	70	147.627	75	219.000	111	141.036	72	125.633	64	121.774	62
XI	43.895	100	43.895	100	43.895	100	49.587	113	59.551	136	76.327	174	47.187	108	45.493	104	61.220	139
VII	40.695	100	40.695	100	40.695	100	41.596	102	53.182	131	58.698	144	41.596	102	52.505	129	54.416	134
XIII	12.500	100	12.500	100	12.500	100	12.500	10	16.291	130	16.291	130	12.500	100	17.725	142	17.725	142
XIV	54.930	100	54.930	100	54.930	100	54.676	100	53.926	98	64.881	118	40.665	74	52.671	96	56.197	102
	33.800	100	33.800	100	33.800	100	51.292	152	48.215	143	60.041	178	39.000	115	43.900	130	48.648	144
XV	22.792	100	22.792	100	22.792	100	24.718	108	22.792	100	22.792	100	24.718	108	27.242	120	27.242	120
	0	100	0	100	0	100	0	100	0	100	0	100	0	100	0	100	0	100
XVI	60.877	100	60.877	100	60.877	100	60.699	100	67.424	111	71.596	118	58.656	96	46.724	77	77.724	128
	409.575	100	409.575	100	409.575	100	341.505	83	350.153	85	384.603	94	293.229	72	454.485	111	511.232	125
XVII	155.485	100	155.485	100	155.485	100	160.137	103	233.608	150	328.433	211	153.880	99	154.964	100	173.486	112
XVIII	208.871	100	208.871	100	208.871	100	212.184	102	232.080	111	268.941	129	208.450	100	201.699	97	198.499	95
	79.052	100	79.052	100	79.052	100	79.052	100	112.679	143	213.518	270	79.052	100	79.052	100	99.449	126
XIX	205.315	100	205.315	100	205.315	100	162.313	79	187.698	91	208.884	101	163.291	80	219.934	107	163.255	207
								97		**127**		**216**		**93**		**104**		**125**

Tabella 4.4 Tabella riassuntiva dei test su vari schemi con velocità v=100m/s e w=10rad/s

4.3 Considerazione

Osservando tutte le tabelle e la media del rapporto percentuale portato sotto ogni tabelle si constata che : nel primo algoritmo non si cambia il tempo totale del percoso aggiungendo dei step, anche come é detto prima non serve prendere più nodi perché alla fine si prende sempre il più vicino, tutto questo dovuto alla regolarità geometrica dei schemi e alla scelta della velocita. per il secondo algoritmo abbiamo visto che aumentando numeri dei step e nodi peggiora, anche non é mai efficiente più del primo algoritmo, questo fatto é causato dai salti che si possono succedere nel'ultimo strato dello schema, che in fatti produce un giro in più,perché tra numero dei nodi più vicini scelti si prende il piu esterno, se sono numerosi il più esterno può essere lontano.il terzo algoritmo come il secondo aumentando il numero dei step e nodi tende a peggiorarsi per le stesse motivi prima citati, solo che con numero dei nodi piccolo ed unico step e più efficiente, visto la media percentuale.

L'effetto della velocita non privilegia nessun algoritmo, come e mostrato nelle tabelle 4.3 e 4.4 abbiamo sempre il 3 algoritmo con unico step é il più efficiente.

Come se può osservare che non é sempre vero che il terzo algoritmo funziona meglio degli altri, ma nel complesso si, in questo caso si può modificare il programma tale che calcola con tutti i tre algoritmo il percorso e dopo andremo a scegliere quello che ha tempo totale minimo, questo scelta chiederà un tempo elevato per lo scopo del progetto soprattutto come ed evidenziato nelle tabelle alle volte anche aumentando numero dei nodi o dei step se minimizza il costo.

Conclusioni

Remane sempre difficile arrivare al percorso ottimo anche usando i tre algoritmi, il fatto di non prendere in considerazione tutti i casi possibile rende l'effecienza bassa. La soluzione ottima sarebbe l'uso di un algoritmo che risolve il problema del commesso viaggiatore come per esempio gli algoritmi di branch and bound utilizzando la programmazione multithreading, pero resta sempre il fatto che il costo computazionale é molto elevato soprattuto per il nostro caso quando si tratta di un scambiatore con numero elevato di curve ad inserire che sarebbe da considerare come nodi il doppio del numero delle curve..

Bibliografia

- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. Introduction to Algorithms, Seconda Edizione. MIT Press and McGraw-Hill, 2001. ISBN 0-262-03293-7. Capitoli 24: Single-Source Shortest Paths, e 25: All-Pairs . Shortest Paths, pp.580-642

- Applegate, D. L.; Bixby, R. M.; Chvátal, V.; Cook, W. J. (2006), The Traveling Salesman Problem, ISBN 0691129932

- Dijkstra, E. W. (1959). "A note on two problems in connexion with graphs". Numerische Mathematik 1: 269–271.

- M. Fischetti, Appunti di Ricerca Operativa, Edizioni Progetto, Padova, 1995.

- D.CATTAI, "SISTEMA ROBOTIZZATO DI ASSEMBLAGGIO PER SCAMBI ATORI", Tesi di laurea, Padova 2010